**Soutenance**

Bonjour et bienvenue à cette présentation. Nous allons donc aujourd’hui discuter de la solution technique retenue pour mener à bien le projet Menu Maker porté par Qwenta.

Notre présentation sera structurée de la manière suivante : nous commencerons par présenter le projet, son fonctionnement et son but ; puis nous verrons la méthodologie de développement retenue ainsi que la gestion du projet en lui-même ; enfin nous verrons les spécifications techniques que nous avons déterminées pour le projet ainsi que la veille technologique qui nous a permis de sélectionner ces solutions.

Je vous propose de commencer dès maintenant en vous présentant le projet. Le but de Menu Maker est de permettre aux restaurateurs d’afficher et de mettre en page leurs menus facilement, en quelques clics. Dans les faits, ce site remplira plusieurs fonctions que voici : pouvoir créer un menu qui comportera un nom, un prix, une description et des catégories de plat ; pouvoir personnaliser son menu, notamment avec un branding restaurateur, la personnalisation du logo, des couleurs de base et de la police utilisée ; pouvoir diffuser son menu via divers canaux ; et enfin l’impression de menu directement depuis le site.

Pour comprendre un peu mieux le fonctionnement du site, voici quelques maquettes. Nous avons donc au départ l’accueil du site (aussi appelé landing page), comportant plusieurs boutons de navigations : le premier dédié aux tarifs, le second à l’espace utilisateur (ou dashboard) et le dernier à des exemples de menus.

Depuis l’espace utilisateur nous avons ainsi accès à la création de menus, à nos menus précédemment créés et à nos informations personnelles. Pour détailler la création de menu, nous avons donc initialement l’ajout de plat ainsi que sa catégorie,

Puis dans un second temps nous avons la personnalisation du menu,

Et enfin la diffusion et l’impression du menu.

Du côté des spécifications techniques souhaitées pour le site, il devra être conçu pour ordinateur uniquement, il n’y a pas de version mobile prévue. Sa charte graphique est également définie : fond beige et blanc, textes en noir, boutons en vert. Pour les navigateurs, nous visons une compatibilité pour Chrome, Firefox et Safari. Le nom de domaine est en cours de validation mais sera très certainement un sous domaine de Qwenta. Pour finir, le site doit être navigable depuis le clavier et lisible par un lecteur d’écran.

Il est temps d’aborder la gestion du projet en lui-même. Pour le mener à bien, nous avons retenu une méthodologie de développement agile, qui est une approche dont l’objectif est de distribuer en continu des logiciels opérationnels créés sur la base d’itérations rapides, aussi appelées sprints. Chaque sprint représente un cycle relativement court durant lequel, l’équipe va devoir réaliser une succession de tâches pour mener à bien une phase du projet. Pour un même projet, différents sprints se succèdent pour, à la fin, obtenir un produit fini. Pour notre projet, nous aurons des sprints de 4 semaines. De cette manière, la prise de risque est amoindrie et l’avancée se fait par étape. Il est alors plus simple d’entreprendre des corrections au fur et à mesure de l’avancée du projet ou d’effectuer des modifications en cours de route, d’adapter les prochains sprints à de nouveaux besoins ou encore, de repenser le projet selon les obstacles rencontrés. Cela permet d’affiner la conformité du projet final avec les attentes du client et d’améliorer sa satisfaction.

Pour mettre en place cette méthodologie, nous nous appuierons sur un outil de gestion de projet, nommé Trello, sur lequel nous aurons notre tableau Kanban, qui permettra de suivre les tâches en temps réel, de gérer les priorités et de favoriser la collaboration entre les membres de l’équipe. Les différentes tâches présentent sur le tableau seront basées sur les « user stories », qui sont des descriptions simples d’un besoin ou d’une attente exprimée par un utilisateur.

Le tableau se présente ainsi, ici nous avons un exemple du premier sprint du projet. Nous avons tout d’abord nos cartes représentant les différentes « user stories », c’est-à-dire les tâches à effectuer qui sont toutes répertoriés dans une même colonne et qui représente notre « Backlog ». Ainsi, pour chaque sprint, nous irons sélectionner dans ce backlog les différentes user stories que nous allons embarquer dans le sprint en cours. A noter que techniquement le backlog ne devrait pas se trouver directement dans le Kanban, mais nous avons dû faire avec les limitations de l’outil Trello. Pour revenir à notre exemple du premier sprint, nous avons donc sélectionné 8 user stories, nous en avons ici 4 qui ont le statut « A faire », 3 qui ont le statut « En cours », et 1 qui a le statut « Test usine ». Il y a également une dernière tâche avec l’état « Terminé », mais il ne s’agit pas d’une user story à proprement parler, mais simplement d’une installation des environnements de développement.

Je vous propose de regarder ensemble une user story en particulier pour mieux comprendre comment cela s’articule. Ici, nous avons l’user story nommée « page login », correspondant donc à la fonctionnalité d’authentification du site. Pour cette user story nous avons donc créé une carte, ou plutôt deux, une pour le FRONT et une pour le BACK, qui contiennent toutes les informations nécessaires au développement de la fonctionnalité associée. Ici nous pouvons voir tout en haut les membres de l’équipes assignés à la tâche, ainsi que les différentes étiquettes correspondant à la priorité et aux domaines de l’user story, suivies d’une estimation de la difficulté de la tâche représentée par un numéro, ce numéro étant ensuite converti en temps de travail effectif. Cette estimation n’a pas été donnée au hasard, elle résulte en effet de ce qu’on appelle un poker planning, où chaque membre de l’équipe va anonymement donner un chiffre représentant pour lui le niveau de difficulté de la tâche. Puis, on met en commun le chiffre de chacun pour déterminer celui qui correspond le mieux à la difficulté de la tâche, c’est-à-dire le chiffre qui revient le plus de fois, tout en essayant de comprendre pourquoi les autres ne l’estiment pas de la même manière. Pour nos poker plannings, nous avons choisi d’utiliser la suite de Fibonnacci, qui est très souvent utilisée et qui a fait ses preuves. Pour revenir à l’user story, nous avons ensuite une description de celle-ci, c’est-à-dire ce que l’utilisateur est censé pouvoir faire, ainsi qu’une section spécifications techniques comprenant des informations supplémentaires pour développer la fonctionnalité. Comme je le disais précédemment, notre estimation de difficulté est ensuite convertie en temps de travail effectif, et nous avons donc ici une estimation de la durée de la tâche, ainsi qu’un outil de suivi. Ensuite, nous avons une ou plusieurs maquettes de la page ou fonctionnalité en question, et enfin les différents critères d’acceptation permettant de valider la complétion de la tâche.

Concernant l’équipe de développement, nous avons prévu 1 développeur Frontend pour développer la partie vitrine du site ; 1 développeur Backend pour développer les coulisses du site (connexion utilisateur par exemple) ; et 1 développeur FullStack pour faciliter les interactions entre Front et Back et venir en renfort sur les tâches les plus ardues.

Et pour la communication, nous préconisons tout d’abord pour l’équipe, des réunions internes quotidiennes d’environ 15mn en distanciel pour faire le point sur la situation, l’avancement et l’assignation des tâches ; des réunions rétrospectives hebdomadaires d’environ 30mn en distanciel pour passer en revue la semaine ; et des réunions bilan à chaque fin de sprint d’environ 1h en présentiel pour présenter les réalisations accomplies et discuter des sprints à venir (implémentation de nouvelles fonctionnalités, modifications de fonctionnalités existantes, corrections de bugs, mises à jour…). D’autre part, pour la communication avec l’équipe, nous préconisons des réunions hebdomadaires d’environ 30mn en distanciel entre le product owner et le client afin de faire remonter les avancées du sprint en cours et les éventuels imprévus ; et des présentations d’environ 1h en présentiel à chaque fin de sprint pour montrer les nouvelles fonctionnalités développées.

Abordons maintenant les choix techniques et technologiques que nous avons pu retenir à l’issue de notre veille. Pour commencer, pour la partie Frontend, nous avons décidé de développer sous ReactJS, qui est une bibliothèque JavaScript utilisée pour construire des interfaces utilisateur. Les avantages de React sont nombreux, notamment sa facilité de prise en main, ses performances et sa flexibilité exceptionnelles, et ses composants réutilisables. Côté Backend, nous avons retenu NodeJS, qui est une plateforme logicielle permettant d’exécuter du JavaScript côté serveur. L'un des avantages de NodeJS est la possibilité d'exécuter plusieurs requêtes vers le serveur simultanément, il n’y a donc pas besoin d’attendre qu’une requête lancée précédemment soit terminée. NodeJS est également très performant et flexible, ce qui en fait donc un choix judicieux pour notre projet. Enfin, pour ce qui est de la partie base de données, nous avons sélectionné MySQL, qui est un système de gestion de bases de données relationnelles. MySQL est très réputé et parfaitement adapté à notre projet étant donné sa simplicité d’utilisation et de configuration.

Concernant les différentes fonctionnalités du site, nous avons fait les choix suivants : pour l’authentification utilisateur, nous utiliserons Nodemailer (librairie Nodejs) pour envoyer un mail de confirmation à l’utilisateur, ainsi que localStorage (objet de stockage web) et PassportJS pour gérer la session de l’utilisateur. Nous utiliserons les librairies React-pdf pour l’exportation en PDF et React-to-print pour l’impression de menus. Nous utiliserons également Redux pour la partie personnalisation de menus. Enfin, nous ferons appel à l’API de Deliveroo pour les fonctionnalités d’exportation et de partage vers cette plateforme. Je tiens à préciser que pour ce qui est d’Instagram, la plateforme ne permet actuellement pas de mettre en place une exportation depuis MenuMaker, c’est pourquoi aucune solution n’est présente ici.

Ensuite, pour la mise en forme du site, nous nous servirons de la librairie Material UI pour les différents composants réutilisables tels que les modales, les cartes et les onglets. Nous nous servirons également de React Hook Form pour les différents formulaires.

Je vais ici détailler un petit peu plus une des spécifications techniques du projet, l’envoi de mail à l’utilisateur via Nodemailer. Tout d’abord voici les maquettes de la page d’authentification, où l’utilisateur va entrer son adresse mail pour s’inscrire ou se connecter et recevoir ensuite un mail pour accéder à son compte.

L’envoi de mail va donc s’effectuer de la manière suivante : nous allons tout d’abord créer ce qu’on appelle un « transporteur » qui va permettre de faire le lien entre nos serveurs et le destinataire du mail, puis on paramètre le message à envoyer (son contenu, son destinataire, son objet, etc.), et enfin on envoie le mail grâce à notre transporteur. C’est une explication assez simplifiée de Nodemailer, mais suffisante pour comprendre le fonctionnement de celui-ci.

Pour terminer sur les spécifications techniques, nos recommandations en termes de sécurité sont les suivantes : l’utilisation de services cloud tels que ceux proposés par Google Cloud Platform ; la mise en place d’un certificat SSL pour sécuriser la connexion ; des sauvegardes régulières du site ; des mots de passes renforcés (recommandations CNIL) ; la mise en place de restrictions d’accès aux fichiers sensibles (administrateurs) ; et se prémunir des failles courantes XSS et injections SQL. Nous prévoyons également le déploiement de correctifs et de mises à jour régulières pour une maintenance optimale du site.

Passons maintenant à la veille réalisée avec Feedly. Feedly est un outil proposant à ses utilisateurs d’organiser leur veille d’information en choisissant les médias qu’ils souhaitent suivre et quel contenu ils veulent lire classé par catégorie. Ici, vous pouvez voir notre veille répartie en 2 axes principaux : un premier centré sur le développement web, c’est-à-dire les frameworks, librairies et bonnes pratiques de sécurité en ligne ; le second centré sur les spécifications techniques associées au projet. Nous avons ainsi retenu environ une dizaine d’éléments pour chaque axe, pouvant venir à la fois de sites web, de journaux ou encore de réseaux sociaux.

Par exemple, pour la partie développement web, nous avons retenu cet article sur les certificats qui permettent de sécuriser son site web, et pour la partie spécifications techniques nous avons retenu cet article sur les avantages d’utiliser ReactJS et NodeJS pour son site.

Les principaux avantages de la veille technologique pour ce projet sont de pouvoir se tenir informé des dernières nouveautés en termes de langages, frameworks, bibliothèques, librairies, etc. ; et de se tenir à jour sur les bonnes pratiques de développement et de sécurité en vigueur, ce qui favorise l’apprentissage continu de nos équipes.

En conclusion, nous avons vu ensemble ce qu’est le projet Menu Maker, décrit la méthodologie, les outils et les spécifications techniques que nous avons retenu pour le mener à bien et enfin l’outil de veille que nous avons mis en place pour développer le site dans les meilleures conditions possibles.

Je vous remercie pour votre attention et reste à votre disposition pour toutes question.